The Spitzer Science Center: using metrics analysis to improve system stability

Susan Comeau, Mark Legassie, Lee Bennett, Suzanne Dodd

Susan Comeau, Mark Legassie, Lee Bennett, Suzanne Dodd, "The Spitzer Science Center: using metrics analysis to improve system stability," Proc. SPIE 7016, Observatory Operations: Strategies, Processes, and Systems II, 701623 (12 July 2008); doi: 10.1117/12.788065

SPIE.

Event: SPIE Astronomical Telescopes + Instrumentation, 2008, Marseille, France

The Spitzer Science Center
Using Metrics Analysis to Improve System Stability

Susan Comeau*[a], Mark Legassie[a], Lee Bennett[b], Suzanne Dodd[b]
[a]Raytheon, 299 North Euclid Ave., Suite 500, Pasadena, CA, 91101
[b]Spitzer Science Center, California Institute of Technology, Pasadena, CA, 91125

ABSTRACT

The Spitzer Science Center (SSC) Software Science Operations System (SOS) is a large, complex software system. Over 1.2 million lines of code had been written for the SOS by time of launch (August 2003). The SSC uses a defect tracking tool called GNATS to enter defect reports and change requests. GNATS has been useful beyond just tracking defects to closure. Prior to launch a number of charts and graphs were generated using metrics collected from GNATS. These reports demonstrated trends and snapshots of the state of the SOS and enabled the SSC to better identify risks to the SOS and focus testing efforts. This paper will focus primarily on the time period of Spitzer's launch and In Orbit Checkout. It will discuss the metrics collected, the analyses done, the format the analyses was presented in, and lessons learned. This work was performed at the California Institute of Technology under contract to the National Aeronautics and Space Administration.

Keywords: Spitzer Space Telescope, software testing, metrics, defects

1. INTRODUCTION

The Spitzer Space Telescope is the final mission in NASA's Great Observatories Program. On August 25, 2003, Spitzer became the largest infrared telescope ever launched into space. Spitzer has an 85-centimeter telescope and three cryogenically cooled science instruments. The Spitzer Science Center (SSC), located at the California Institute of Technology, is responsible for science operation including proposal calls, observatory scheduling, data processing, and data archiving. In order to accomplish this task the SSC has developed a complex Science Operations System (SOS). Creation and maintenance of the SOS required development of approximately two million lines of code.

In 2000 the SSC began using a problem reporting tool, GNATS, to track defects and request changes to the SOS software. GNATS is primarily used to keep track of known problems with the SOS, request new functionality in a SOS sub-system, and assign fixes and changes to builds. The introduction of GNATS allowed the SSC to begin collecting metrics such as the amount of changes requested to SOS sub-systems and the number of defects found in the sub-systems. As the project approached launch it became increasingly important to confirm the stability of the SOS and prioritize testing. Analysis of metrics assisted in these activities.

This paper is an examination of the metrics gathering and analysis efforts at the SSC, primarily at the time shortly before Spitzer's launch and during the In Orbit Checkout (IOC).

2. METHODOLOGY

2.1 GNATS Defect Tracking Tool

Early on in the project the SSC System Engineering and Integration and Test (SEIT) team realized using a defect tracking tool would be a critical part of building a stable SOS. The freeware tool GNATS[1] was selected because it had the required minimum functionality (with minor modifications); the tool allowed the team to create reports of defects found in the SOS, request changes to the SOS, and assign fixes and changes to builds. A report created in the GNATS tool is called a Problem Report (PR). Figure 1 is diagram of the lifecycle of a PR.

*susan@ipac.caltech.edu; www.spitzer.caltech.edu

[1]www.gnu.org/software/gnats/

Observatory Operations: Strategies, Processes, and Systems II, edited by Roger J. Brissenden, David R. Silva, Proc. of SPIE Vol. 7016, 701623, (2008) · 0277-786X/08/$18 · doi: 10.1117/12.788065

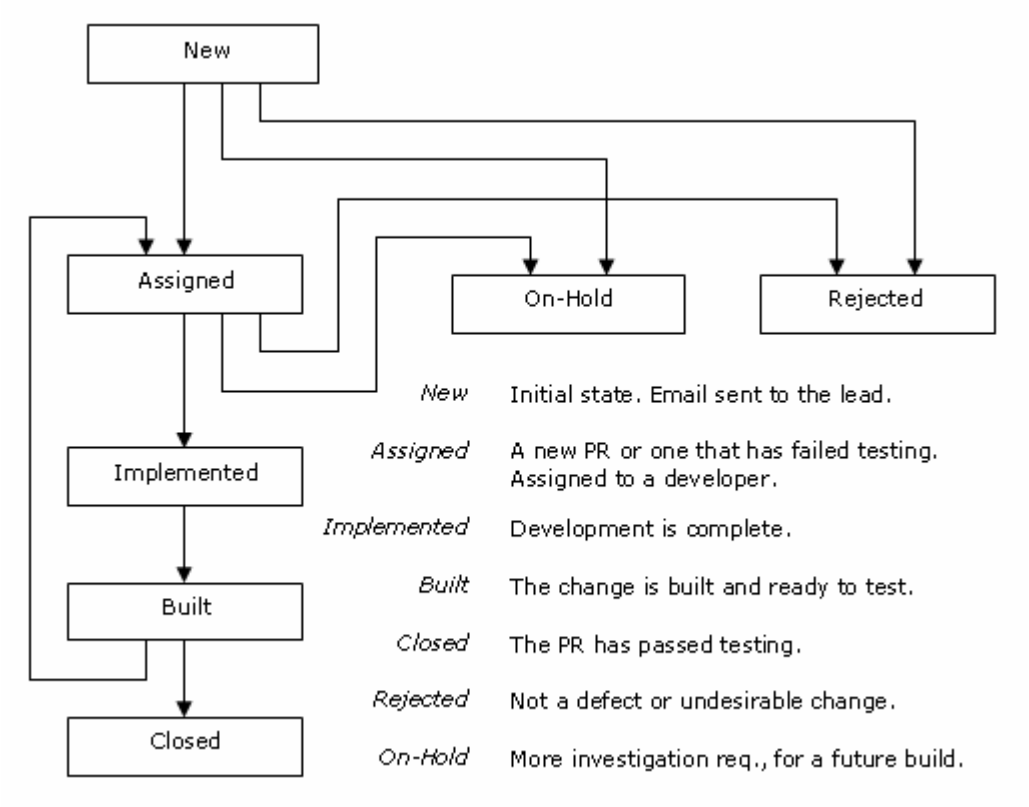

Figure 1: Life-cycle of a Problem Report in GNATS.

A PR in any state other than "Rejected" or "Closed" is considered an open issue with the SOS.

All PRs in GNATS start in the "New" state. When a PR is entered an email is automatically sent to the team lead for the appropriate SOS sub-system. Each PR is marked as either a Change Request (CR) or Anomaly Report (AR). CRs are requests for new functionality while ARs are defects. If a lead feels an AR is not a real defect (it is a user error, for example) he or she may move the PR to the "Rejected" state. Otherwise the PR is generally moved to the "Assigned" state and assigned to a particular developer. The SSC has a weekly Change Control Board (CCB) meeting to discuss open PRs and upcoming builds. PRs in the "New" or "Assigned" state are assigned to a particular Target Release at that time. The Target Release of a PR is the version number of the SOS the code will be built in. If the CCB feels that a requested change is not desirable that PR is moved to the "Rejected" state. If a change is deemed desirable, but will not be worked on in the near future or needs further investigation, it is moved to the "On-Hold" state.

Once a PR is assigned to a Target Release the coding is begun. The PR moved to the "Implemented" state when the work is complete and checked into source control. The PR remains in the "Implemented" state until the build has been delivered to the SEIT team for testing. At that time it is moved to the "Built" state by the Configuration Management lead. This indicates it is ready to be tested.

During a test cycle the SEIT team assigns PRs to particular test cases and verifies the fix was made or the requested functionality was added. If a PR passes testing it is moved to the "Closed" state. Otherwise, it is moved back to the "Assigned" state where eventually the CCB will designate it for a different Target Release. While a PR is in any state other than "Rejected" or "Closed" it is considered an open issue with the SOS.

When a PR is created it is assigned a unique identifier which allows users to find it and update it. Table 1 lists the fields that can be filled out when the PR is created. The fields which are most important when tracking an individual PR to closure are not necessarily those that are most useful when gathering metrics since metrics analysis is concerned with long term trends.

Table 1: Information entered in GNATS when creating a PR.

Field	Use
Reporter's Email	Email address of person who submitted the PR.
CC Email List	List of email addresses to copy on updates to the PR.
Category	SOS Sub-system the PR applies to.
Synopsis	One line description of the defect or requested functionality.
Urgent	Not used.
Criticality	1 through 4. Criticality 1 indicates a serious risk to Spitzer operations. (Added by modifying the GNATS source code.)
Priority	Not used.
Class	Change Request or Anomaly Report
Submitter-Id	Not used. (Always set to "SSC".)
Originator	Originator of the PR. (Could be different than the reporter if originated by an outside user.)
Release-Found	Build version number the defect was found in.
Target-Release	Build the change will be incorporated into. (Added by modifying the GNATS source code.)
Description	Detailed description of the defect or requested change.
File Attachments	Additional material describing the defect or requested change.
How-To-Repeat	Steps that can be repeated to demonstrate the defect.
Fix	Filled in by the developer with what was done to fix the defect or add the requested change.

2.2 The Need for Metrics Collection and Analysis

As the Spitzer launch date approached part of the SEIT team's preparation for IOC was proving the Level 3 SOS requirements had been met and passed testing, demonstrating the SOS was stable and had no serious defects that would prevent science operations, and devising a process for doing rapid builds during IOC.

In order to prove requirements compliance the SEIT team went through the SOS Requirements from the Science Operation Requirements Document (SORD) and verified each requirement was assigned to a test case. A matrix was created to trace the requirements to test cases. For each requirement the table listed the associated test case, the status of the test case (Pass/Fail/NA), and any PRs created as a result of executing the test case.

Before launch the SSC performed fewer than four major builds a year. During IOC, however, the plan was to do one build every week in order to rapidly fix any defects found. This meant that ARs had to be entered immediately and the CCB had to carefully prioritize the ARs and assign them to builds. For the SEIT team, the significant change from the major builds was to determine where to focus testing resources in order to rapidly, but sufficiently, test builds.

In an effort to help all of these activities GNATS PR metrics collection was begun. The remainder of this section will describe the metrics which were collected and the format in which they were reported or presented.

2.3 System Release Review

For each SOS build a System Release Review (SRR) meeting is held to present changes to the SOS, test results, and deployment information to the stakeholders. The SRR is attended by a representative from each team at the SSC, SCC management, the JPL Mission Manager, and representatives from JPL Quality Assurance and Configuration Management. At the SRR management decides if the build will be accepted and deployed based on the information

presented in the presentation. Five reports presented in the SRRs shortly before launch will be discussed below. The reports were generated using metrics collected from GNATS by the SEIT team.

The staffing resources required to create the reports were approximately one full time equivalent (FTE) for several days and the assistance of a part-time documentation person. It was determined to be to the benefit of the project to devote those resources to generation of these slides.

SSC AR Status by Software Release

This report consisted of a table listing each build version of the SOS and giving the number of ARs which were New, Closed, Open, or Awaiting Verification at the time each build was released. Only Anomaly Reports were counted, not Change Requests.

For this report the New column totaled all ARs opened since the previous build's release date. Open totaled all ARs not in the "Closed" or "Rejected" states. Awaiting Verification totaled the ARs in the "Implemented" or "Built" state. Open totaled the remainder of the ARs. Closed was the sum of Open ARs on the previous build's release date plus the New ARs for this release minus the Open ARs for this release. Table 2[1] gives an example of this report.

Table 2: Example of AR Status by Software Release Report

Release	Release Date	New	Closed	Open	Awaiting Verification
S6.0	04/15/2002	258	163	234	0
S6.1	05/26/2002	260	123	381	0
S6.2	08/30/2002	113	155	132	196
S6.3.1	11/22/2002	149	211	127	137
S6.3.3	01/08/2003	51	150	119	46
S7.0.0	03/14/2003	122	143	62	81
S7.1.1	05/14/2003	96	118	61	60
S7.5.1	06/09/2003	37	38	57	56
S8.0	07/30/2003	92	84	59	68

This table shows some interesting trends. Starting with the S6.2 the SEIT was unable to test and close all ARs in GNATS by the time of the SOS deployment. It is normal to have several ARs targeted for a future build Awaiting Verification at a SOS deployment, but the total of 196 is much higher than desired. This indicates the number of changes incorporated in the SOS at that time may have been greater then the SEIT team had resources enough to handle.

SSC Sub-System Stability-Over-Time

For the SRR presentation a graph was made demonstrating the stability-over-time of each SOS sub-system. These graphs were made by plotting the total number of CRs for the sub-system over time. The thinking was that large number of changes to a system indicated the system was not becoming more stable over time. Figure 2[1] shows the Stability-Over-Time graph for the SOS Uplink sub-system Spot. (Spot is the tool used by astronomers to plan observations and submit proposals.) Figure 2 shows that as of August 2003 approximately 300 functionality changes had been requested in the Spot sub-system. It is expected that as a sub-system becomes stable eventually the number of changes requested will plateau.

Figure 3 shows the Spot Stability-over-time graph extended out to May 2007. The graph shows that the number of functionality changes requested in Spot has not slowed over time. This information can by used by management to when assigning development resources. It is also extremely useful to the SEIT team, which uses risk to prioritize

testing. Sub-systems with many changes may have had more defects introduced and might need more testing resources assigned to them.

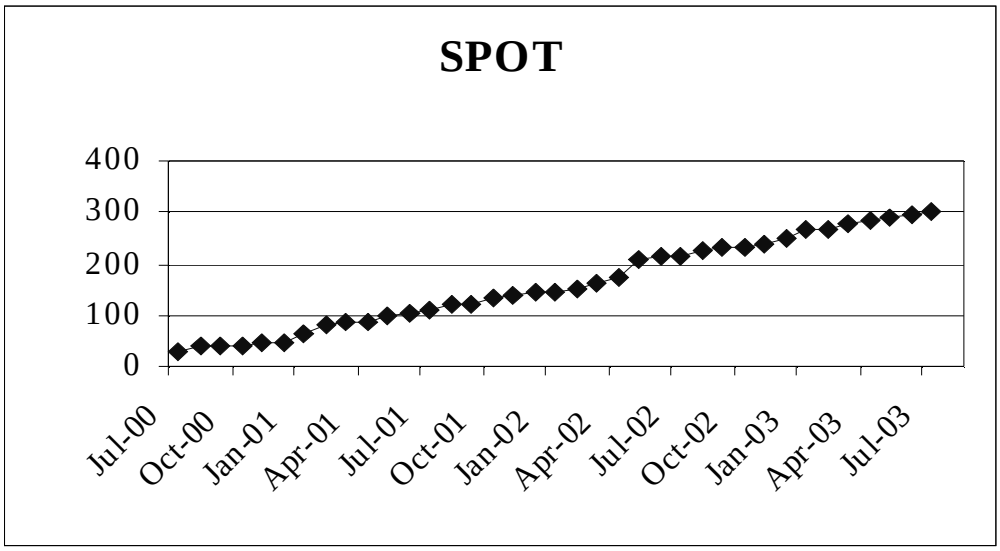

Figure 2: Example of Spot Stability-Over-Time Graph

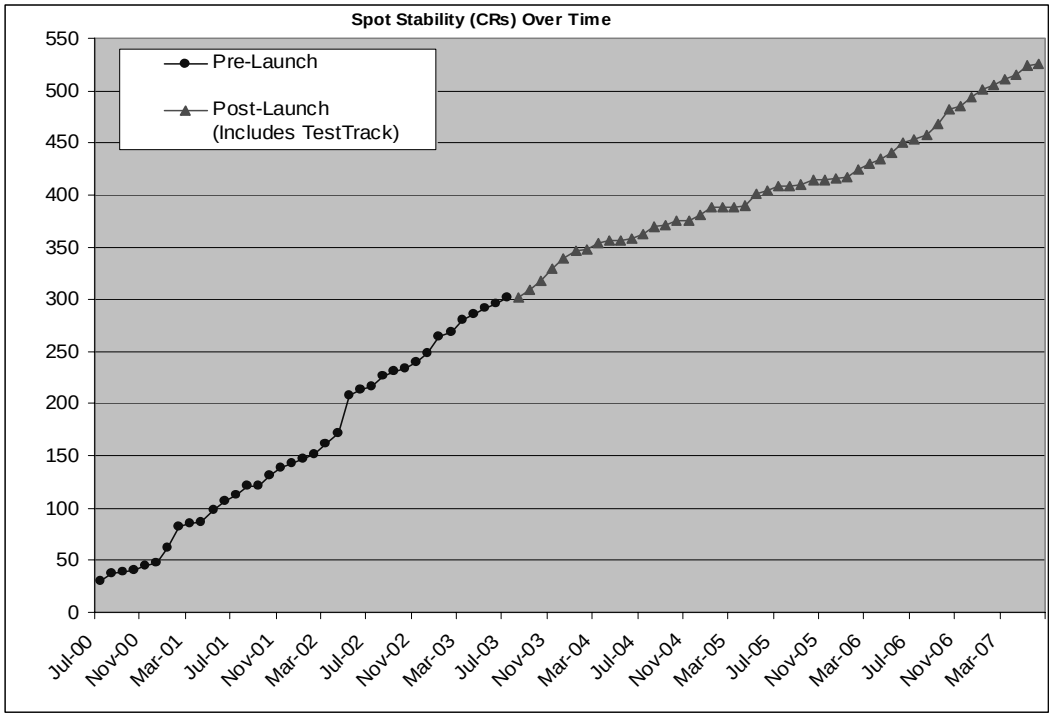

Figure 3: Spot Stability-Over-Time Graph extended through May 2007

SSC Sub-System Anomalies-Over-Time

For the SRR presentations before launch a graph was made demonstrating the defects over time in each SOS segment (Uplink, Downlink, and Science Database Management). These graphs were made by plotting the total number of ARs for the segment over time. Figure 4[1] shows the total ARs by sub-system in the Uplink segment, broken out by sub-system. Figure 5[1] shows the total ARs further broken down by criticality.

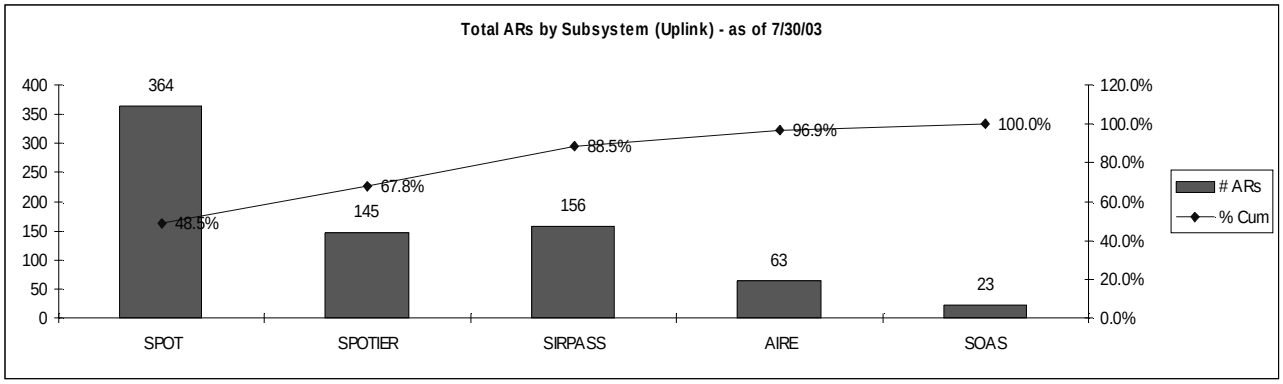

Figure 4: Example of Uplink Anomalies-Over-Time Graph (broken out by sub-system)

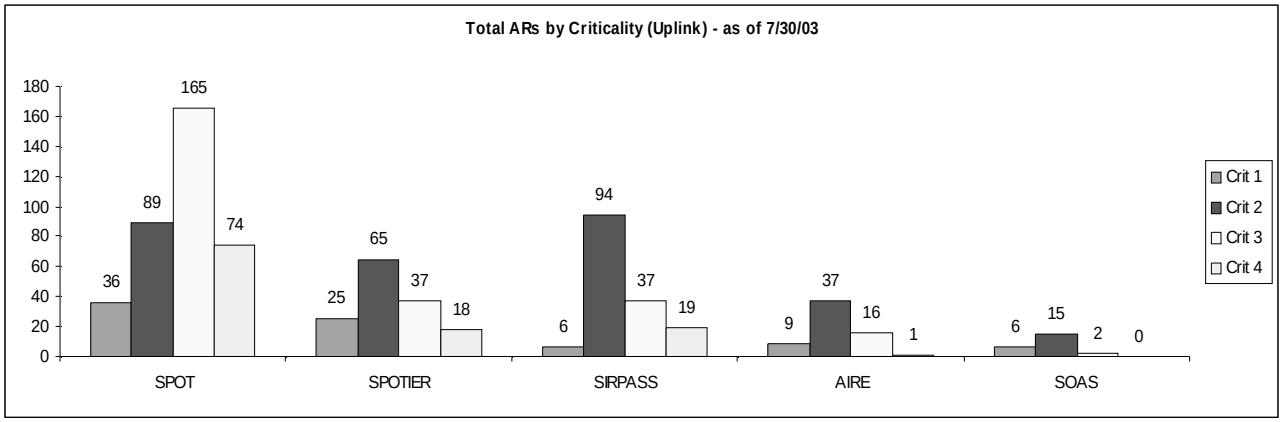

Figure 5: Uplink Anomalies-Over-Time Graph (broken out by criticality)

The type of information imparted by these graphs is extremely important to the SEIT for risk based testing. From Figure 5 it is appears that at that time Spot required more testing resources than AIRE because of the number of defects found in the sub-system. What is not evident from the graphs, however, is that the AIRE sub-system is in the SOS critical path and therefore the changes allowed in the sub-system were more tightly controlled. Even though AIRE had fewer defects found, when testing resources were being assigned it was more important to the mission to have AIRE thoroughly tested than Spot. Metrics analysis can highlight areas of concern in a project, but it doesn't necessarily present the entire picture.

SORD Requirements Trace

Since before launch, the SOS Test Report for every major build has contained a matrix tracing the SSC Level 3 requirements to test cases. The table has row for every SORD item, identified by paragraph number. The following columns are filled are out after every test cycle: Test Case number, Test Result (Pass/Fail/NA), ARs/CRs Created, Date Tested, and Comments. This matrix allows the project to easily verify each of the SOS requirements was implemented and tested. Figure 6[1] is a slide from the S8.0 SRR presentation which summarized the requirements testing results.

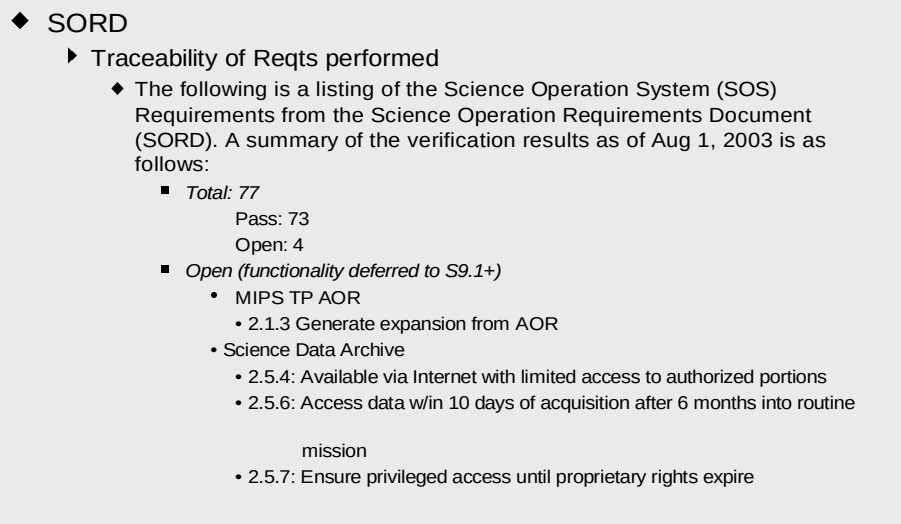

Figure 6: Example of SORD Requirements Trace Summary

SOS Line of Code Count

For the SRR presentations before launch a graph was made demonstrating the lines of code in the SOS, broken out by segment (Uplink, Downlink, and Science Database Management). Figure 7[1] shows an example of this graph. The SOS had about 1.3 million lines of code at launch and has approximately 2 million lines of code at the current date.

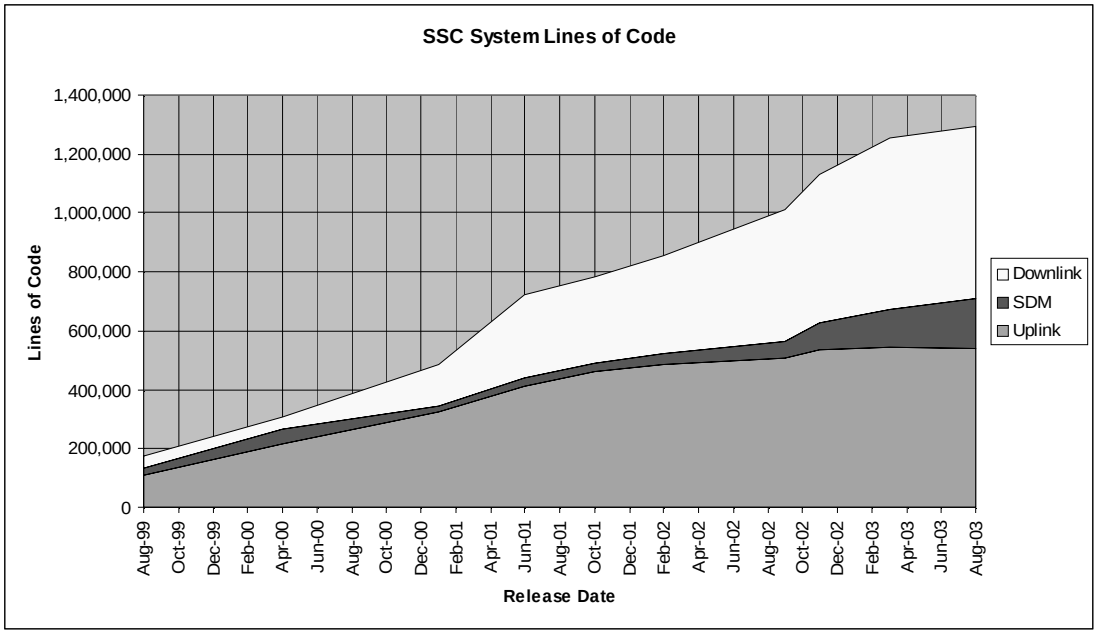

Figure 7: Example of Line of Code Count Graph

2.4 Weekly Reports

During IOC the SSC began doing weekly builds of the SOS to rapidly fix any serious defects found. Weekly reports were created for SSC management generated from PR metrics from GNATS collected by the SEIT team. These reports were intended to keep management apprised of the state of the SOS. Two of those weekly reports will be discussed below. The reports were created by a part-time documentation person.

Weekly SSC Anomaly Report Status

Figure 8[2] demonstrates the number of PRs that were New, Close, Open, and Awaiting Verification over time.

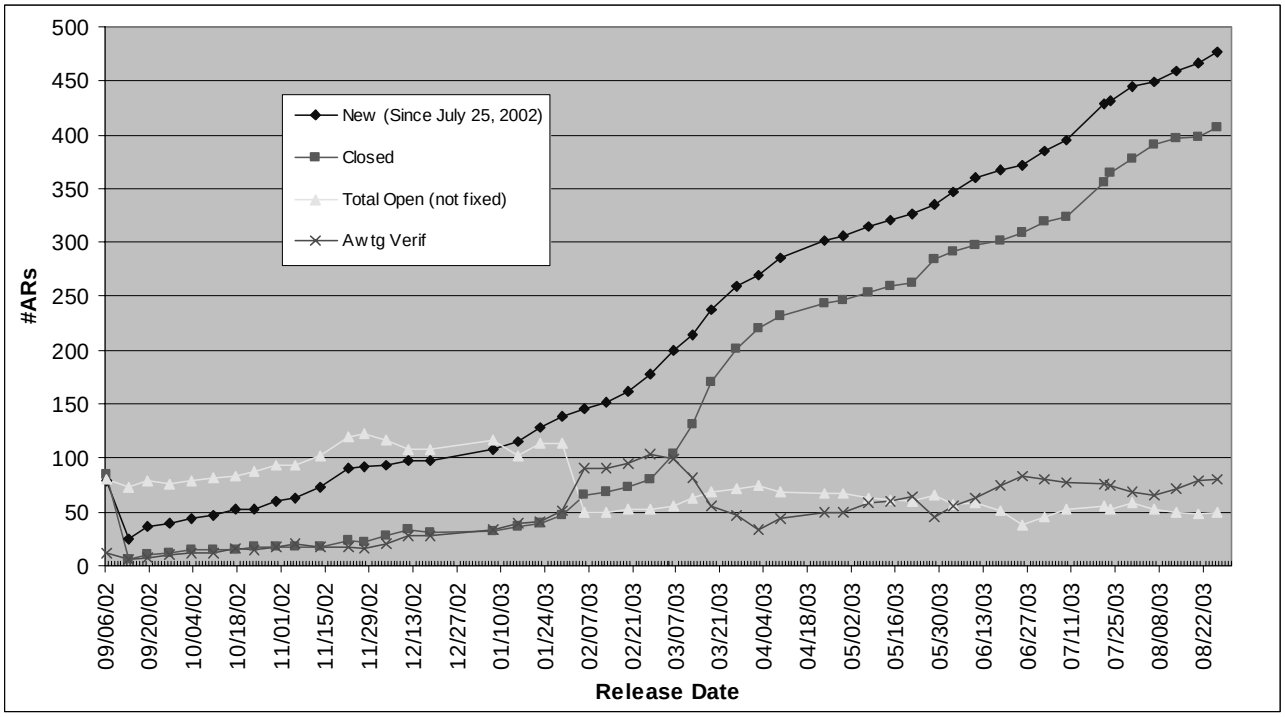

Figure 8: Example of SSC Anomaly Report Status Graph[1]

The Closed line shows the total number of defects that have been fixed, tested, and released. It is expected that not all ARs would be closed immediately because lower criticality problems are targeted for builds that could be six months or a year in the future. An example of a criticality 4 AR is a spelling error in a Help window. It is a real problem, and should be fixed some day, but there are higher priority defects that should be fixed first.

SSC Anomaly Report Status

Tables 3 and 4 are from the weekly report generated on August 26, 2003 (one day after launch). Table 3 contains all open ARs, broken down by criticality and segment. Table 4 shows all ARs awaiting verification, also broken down by criticality and segment. These two reports give a snapshot of the SOS. The things management would be looking for would be: no criticality 1 ARs, and the criticality of any open ARs in the Uplink or SDM segments.

Table 3: Example of AR SSC Anomaly Report Status All Open ARs Table

Segment/Criticality	Uplink	Downlink	SDM/OPS	Other	*Total*
1	0	0	0	0	*0*
2	4	2	4	0	*10*
3	13	4	5	1	*23*
4/NA	13	2	0	1	*16*
Total	*30*	*8*	*9*	*2*	*49*

- *Crit 1 Open: None*

Table 4: Example of AR SSC Anomaly Report Status Awaiting Verification Table

Segment/Criticality	Uplink	Downlink	SDM/OPS	Other	*Total*
1	0	0	0	0	*0*
2	4	21	7	2	*34*
3	2	27	13	1	*43*
4/NA	0	2	0	1	*3*
Total	*6*	*50*	*20*	*4*	*80*

Along with the information shown in Tables 3[2] and 4[2], the weekly reports also listed the ARs scheduled for the next two weekly builds. The slides in Figure 9[2] are also from the weekly report of August 26, 2003.

Open S8.1 ARs:			
4712	2	downlink-pipeline-exec	issues with updateCalExposureTypesandPlScriptIds
4711	2	ops-database-tools	archtool - accessprogram routine cannot be resolved
4715	3	ops-database-tools	dbtool - timing window durations, start/stop times in units of seconds
4661	3	sos	S8.0 Ops Checkout: FTZ not set up for some IST accounts
Open S8.2 ARs:			
4574	2	uplink-sut	Proposal submission: Program name error when submitting proposal update
2319	2	uplink-sut	"proposalfile" column getting cleared by Proposal Tool update
4700	2	sdm-proposal	Modify the stored procedure proposalLogin to accept role "spot_read"
4646	2	ops-database-tools	SetSchedWeight tool fails for "obscomment" option
Awaiting Verification S8.1 ARs:			
4615	2	downlink-mips-ge	interp module fails to interp properly
4611	2	downlink-mips-ge	Duplicate entries for pipenum 346 in plinexyz.cdf
4593	2	downlink-pipeline-exec	MIPS Dark plscriptid's need distinct priorities
4590	2	downlink-mips-ge	MIPS dark combine now fails
4578	2	downlink-mips-ge	pointing thread fails due to plscripts table
4562	2	downlink-ipm	hkloader error during ingest of HK data (value exceeds limit of integer precision)
4665	2	ops-database-tools	CAVE: unable to run it in Ops proposals database
4623	2	sdm-database	The stored function smallFloatToInterval fails on certain values.
4581	2	sdm-database	no entry in filetypes for memoryDumps
4552	2	ops-database-tools	pmtool fails to X reference new observer id into program
4580	2	sos	Downlink S8.0 IRAC won't work properly with S8.0 Uplink due to HDR change

Awaiting Verification S8.1 ARs: (Cont)			
4663	3	downlink-pipeline-prod	bug in mopex.pl with naming interpolated uncertainty image
4641	3	downlink-irs	problems with crosstalk in 7.5/S8.0 version: patch imminent
4639	3	downlink-irs	Sensitivity to NaNs in SLREMOV discovered in S7.5 Online pipeline tests
4635	3	downlink-ipm	mean value greater than max value
4613	3	downlink-pipeline-prod	fix a bug in sourcestimate that causes array access violation for a stack of > 1000 images
4583	3	downlink-pipeline-prod	in sourcestimate move the bail out branch into the right place
4582	3	downlink-pipeline-prod	in dlcs library fixed copying of the cd-matrix elements from the wcs structure
4510	3	downlink-mips-ge	calkeywords not invoked for MIPS160 illumination correction
4678	3	sdm-database	Select permissions
4671	3	sdm-database	No role set to run certifyCampaign (and others)
4636	3	ops-database-tools	archtool - bugs and suggested changes
4631	3	ops-database-tools	pmtool - bugs and some suggested changes
4554	3	ops-database-tools	dbtool: minor bugs
4174	3	sdm-database	OPS databases staging and archive have tables and procedures that should be removed.
3857	3	sdm	FTZ tool - grep error
4645	3	operations-pipeline	Ensemble creation doesn't update APES master list
4566	4	downlink-pipeline-ing	Some ecsv data is not getting deleted on FEI server after Ingest
Awaiting Verification S8.2 ARs:			
4709	2	uplink-sut	SPOT: auto-update requires two logins
4694	2	downlink-mips-ge	channel 3 darks ensemble fails with 64
4691	2	downlink-irac	plscriptid 27 fails in pointingrefine module
4600	2	ops-database-tools	aorAcc tool failing for ATLO LF1+ AARs
4697	2	archive	FTZ java API is sending the wrong "bye" command
4706	3	downlink-pipeline-ing	reruningest.pl failure
4699	3	downlink-mips-ge	stacklayer does not work correctly for 160 mu
4716	3	ops-database-tools	dbtool - View -> view request window very jumbled
4701	3	sdm-database	Correct error in database procedure releaseRequest

Figure 9: Example List of ARs Scheduled for Upcoming Builds

The ARs in the "Open" list in Figure 9 had been assigned to the S8.1 and S8.2 builds, but the coding had not been done yet. The ARs in the "Awaiting Verification" list were implemented and checked into source control. During IOC the CCB approved every AR that was assigned to a weekly build.

2.5 Quarterly Reports for JPL QA

Before launch the Jet Propulsion Laboratory Quality Assurance group required the SSC to generate quarterly reports giving the number of ARs open greater than ninety days. It was not trivial for the SSC SEIT team to calculate this data since GNATS was not set up to do that type of query. Table 4 shows the information that was collected for these reports.[3]

Table 4: Example of an SSC Quarterly Report to JPL QA

On this date:	**11/30/01**	**12/31/01**	**1/31/02**	**2/28/02**	**3/31/02**	**4/30/01**
Total ARs created (since 7/10/00):	545	566	606	691	766	861
Of ARs currently open, # open > 90 days:	66	68	96	101	85	76
Total ARs closed (since 7/10/00):	401	421	426	476	543	607
Of total ARs closed above, # open > 90 days:	161	168	169	194	220	242
Of total ARs created above, # open > 90 days:	227	236	265	295	305	318

3. RESULTS

The graphs and tables listed under the SRR section continued to be generated for several releases after IOC. As time went on, however, the number of FTEs in the SEIT group was reduced by nearly half and the team's part-time documentation person left the project. Metrics collection and documentation efforts had to be reduced to allow staff to focus on testing. Many of the reports are no longer generated. Before launch and during IOC, however, metrics analysis was very helpful for the SEIT team in focusing testing efforts for risk-based testing. The SOS performed well during IOC, with no major defects found in the critical path.

The SEIT team still does informal metrics analysis when planning a test cycle. While no graphs or charts may be generated, the team still looks at what SOS sub-systems contain the high criticality ARs and which sub-systems contain the most CRs. More testing time is spent on those sub-systems.

Long term, it would have been useful to continue generating the system stability graphs for each sub-system. As the graph in Figure 3 shows, several years after launch the changes requested in some sub-systems is not leveling off. This information could be used to make staffing plans or as input to a discussion of when change requests should be frozen for the various sub-systems.

4. CONCLUSIONS

4.1 Staff Buy-In

For various reasons it can be difficult to convince people to create PRs. Developers sometimes don't see the need to create a change request to add a small modification to the code. Sometimes development group leads don't like to see defect reports written against software they are responsible for. People in general resist doing activities that are seen as paperwork with no value added. In order to get staff buy-in everyone must understand what the purpose of the process is. Defect reporting should not be used as an input to performance reviews, since that merely encourages people to not write defect reports.

4.2 Defect Tracking Tool

As a free tool, GNATS does not have much reporting capability built into it. Future projects could reduce staffing resources for metrics collection and analysis by selecting a defect tracking system that had better reporting functionality. Considering the staff time required to perform the metrics collection over the years, it might have been worth the extra expense to purchase a system with reporting capabilities or to have written scripts to save effort.

As Table 1 shows, there are three fields in a GNATS PR which are not used by the SSC. Future projects should consider carefully which fields are really necessary in a defect report.

4.3 Metrics Usage

It is useful to begin metrics collection early in a project. That way a baseline can be established and changes over time can be tracked. The SSC should consider resuming generation of some reports such as the System Stability Over-Time graphs.

The "Category" field in GNATS is used to track the sub-system the change or defect applies to. In the Uplink segment, for example, there are many sub-systems such as Spot, SIRPASS, etc. Using the category field allowed the SSC better insight into which particular area problems were being found in.

Tracking the number of ARs open longer than ninety days did not prove useful for the SSC. The SSC build cycle is such that even if a defect was fixed in the release immediately prior to the one in which it was found, it may have been open longer than ninety days. Setting importance on an arbitrary date for closure discourages properly prioritizing defect fixes. It doesn't make sense to close a criticality four AR sooner than a criticality two AR simply because the first has been open longer.

Analyzing metrics becomes difficult if changes are being introduced to a set of code that is used for multiple projects. The Uplink sub-system Spot has core code that is now being used by other missions. The changes requested by other missions are being entered in a separate defect tracking system from GNATS. In order to collect metrics for this sub-system the SEIT team now needs to combine data from two different defect tracking systems.

Metrics collection is very useful for risk based testing. When there are not enough testing staff resources to completely regression test a release, determining what sub-systems have risky changes is critical when planning a test cycle.

Before beginning to collect metrics determine what the objective is and what kind of collection and analysis needs to be done to meet that objective. This can help determine what kind of defect tracking system is necessary and what staffing resources will be required.

REFERENCES

[1] Heinrichsen, I., "S8.0 System Release Review Presentation", (2003).
[2] Heinrichsen, I., "SEIT Weekly Report August 26, 2003", (2003).
[3] Eyre, J. "SSC Quarterly Report to JPL QA April 30, 2001", (2001).